dDOSI Spectrum Analysis Unit

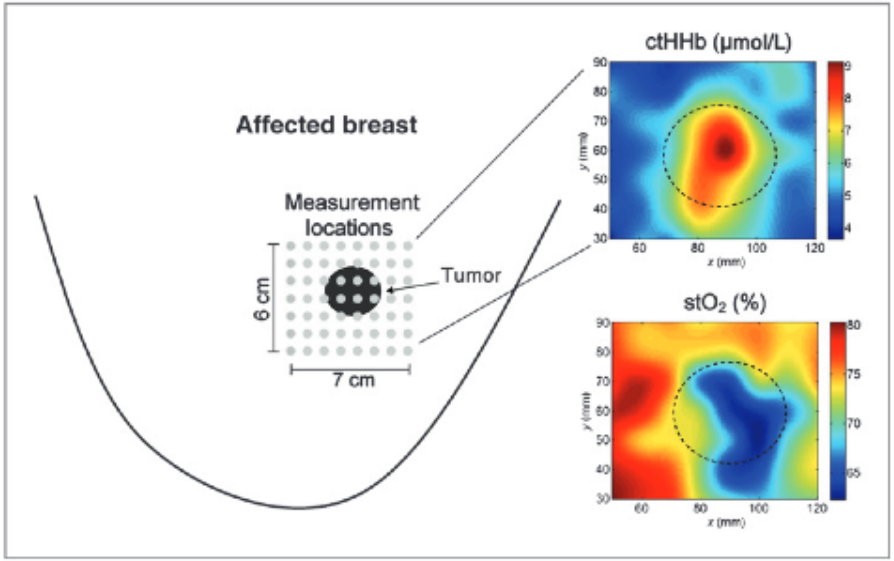
Preliminary Design Review

dDOSI

Team dDOSI (#19)
Caroline Ekchian, Benjamin Havey, Andy Mo,
Thomas Nadovich, & Chris Woodall
Client: Darren Roblyer

Intro - Objective

The dDOSI Spectrum Analysis Unit (dSAU) generates reference waveforms and records the resultant phase shift and amplitude degradation after passing through a sample of human tissue.



* Image from [1]

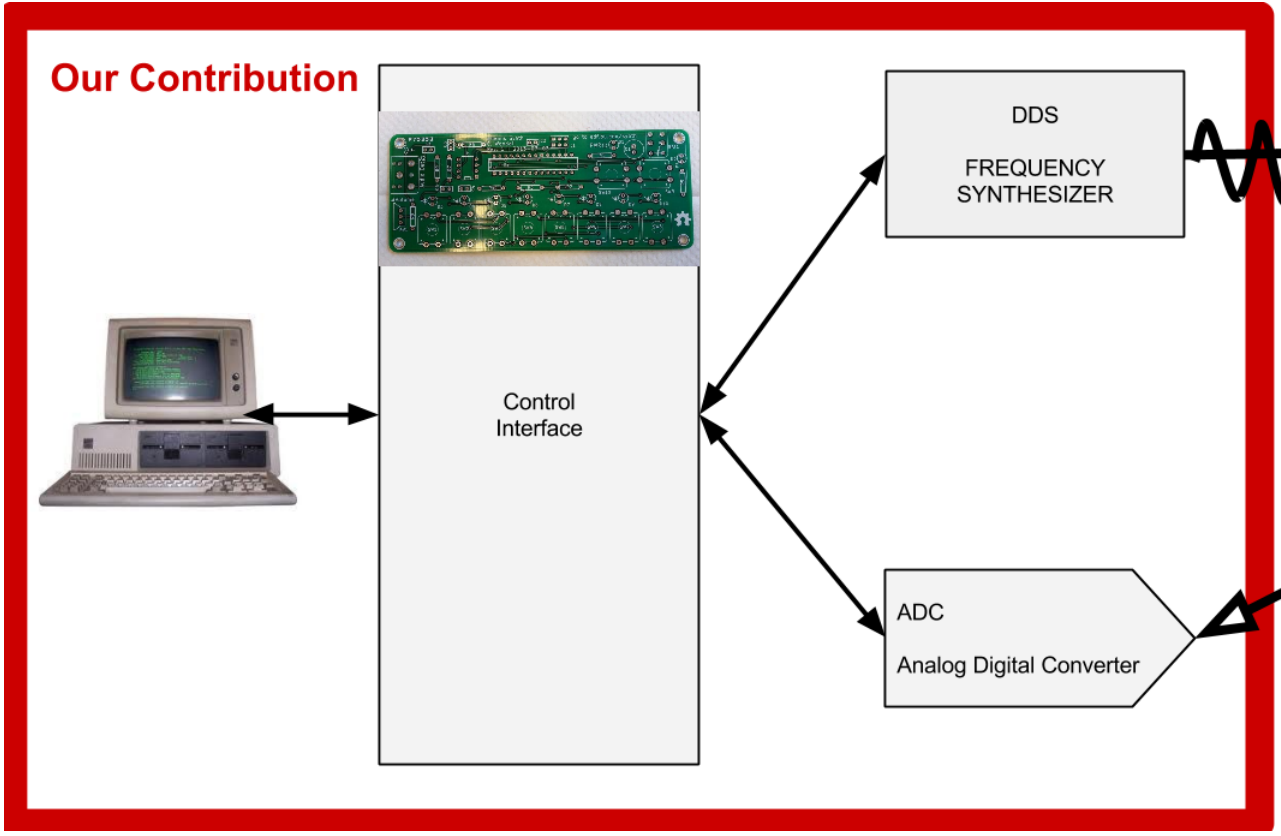
Intro - Background

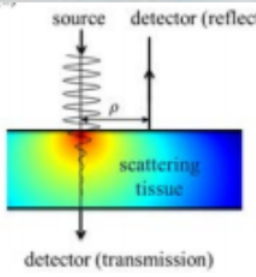
- Our client is designing and building a digital diffuse optical spectroscopy imaging device to advance his research into the applications of diffuse optics to human tissue analysis, specifically cancer diagnostics research.
- Diffuse Optical Spectroscopic Imaging has been proven to be effective at determining concentrations of key molecules (such as water and lipids)
- This technology would provide faster, safer and less expensive tissue imaging than solutions currently on the market.
    - Suggested to be used as a supplemental imaging modality for determining effect of chemotherapy on tumor.

Intro - Background

- Dr. Robyler has built two prototype systems:
  1. Utilizes a Network Analyzer to modulate the lasers and measure the phase and amplitude modulation.
  2. Utilizes a set of development boards, including a 1.8GSPs ADC, to prove the feasibility of direct digital sampling.
- Current systems are functional, but are either extremely expensive or too slow for many potential applications:
  a. Continuous monitoring during chemotherapy
  b. Continuous monitoring during heartbeat
- We are designing an integrated solution to generate and measure waveforms, and interface with a host computer for data transfer and control; a dDOSI spectrum analysis unit

Visualization





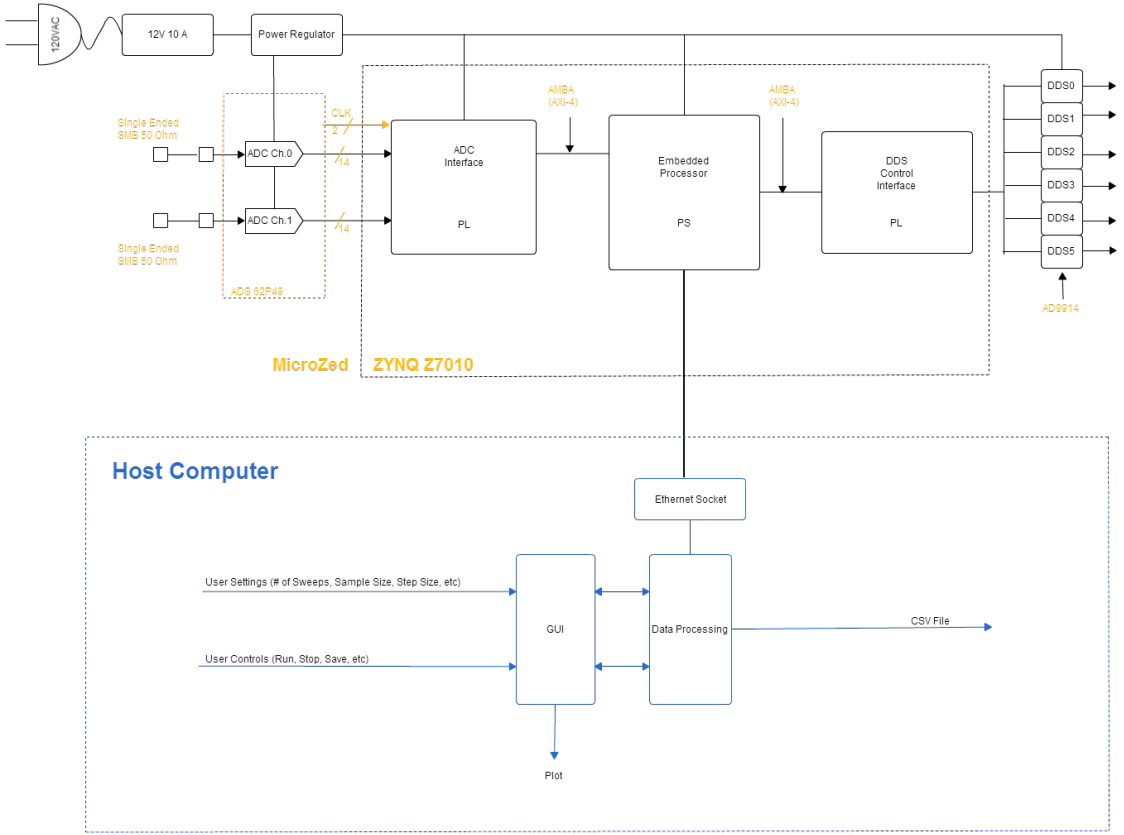
Requirements I - Signals

- Read a 500 MHz signal (50 Ohms single ended input)
- Generate 6 channels of frequency sweep over a range of values between 50 MHz and 500 MHz in steps of a minimum of 1 MHz to drive 50 Ohm single ended outputs from 0dBm to 4dBm
- Independently modulate and update all 6 channels at the same time.
- Complete frequency sweep in less than 2 seconds
- Noise floor above -80dBm

Requirements II - Software

- Communications line between computer running software and PCB running firmware/software
- Display visualization of results and record data into easy to process format.
- Sample size must be adjustable up to 16kSamples/step
- Set number of samples, step size and other configurations in a GUI
- Provide a Windows DLL file for further software development
- Provide full design and usability documentation

Level 1 Block Diagram

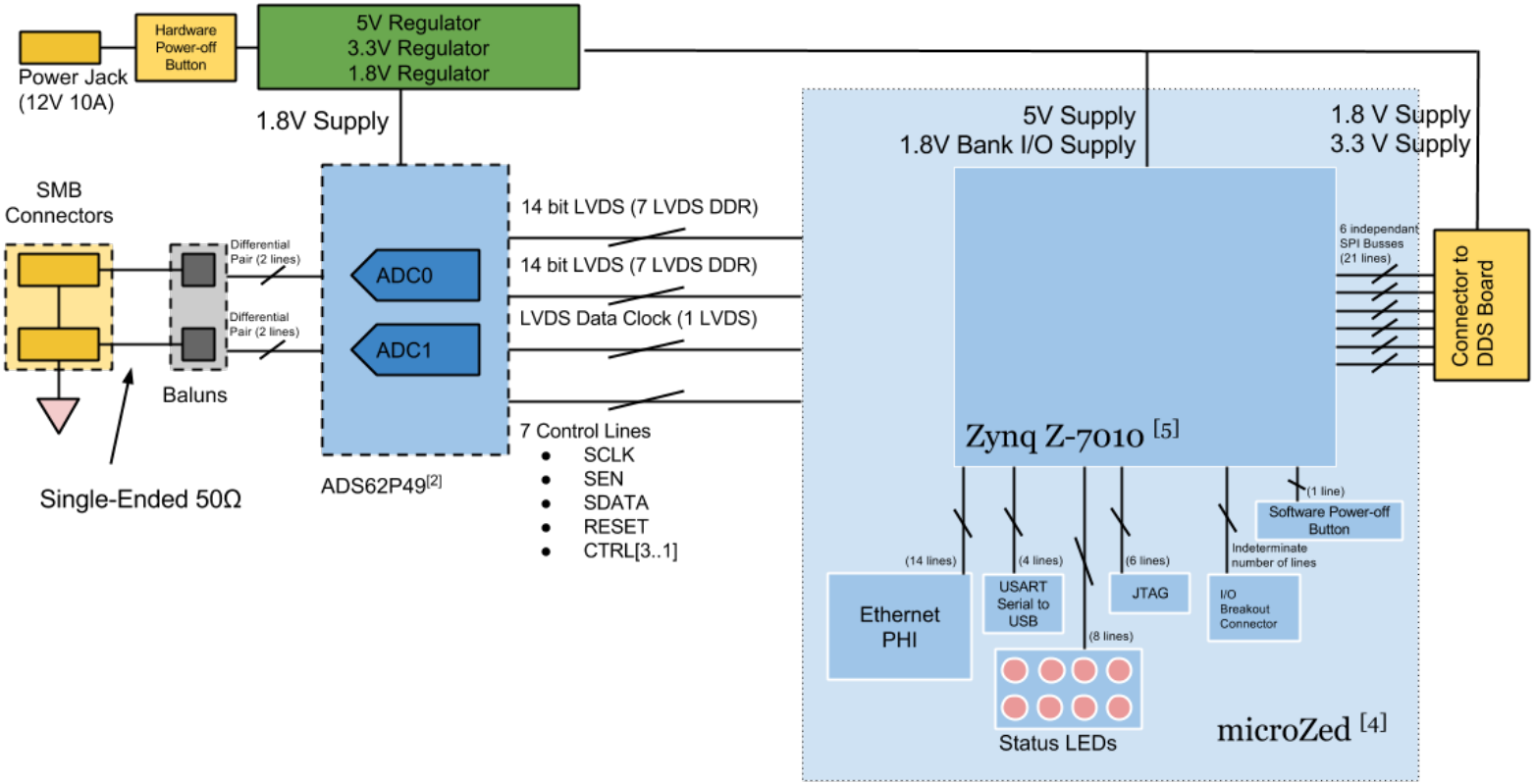


Design Overview

- Modulation signals generated from an array of six AD9914 DDS chips
- Return signals are sampled with ADS62P49 ADC
- Control logic and communications provide by the Zynq 7Z010 SoC on a microZed development board
- 1000 Mbps Ethernet connection to local PC running Windows
- Software on the PC will be written in Visual C++
  - GUI for executing tests
  - Preliminary visualization of data
  - Data recording in .csv file format

Hardware Design

Hardware - Motherboard Block Diagram



Hardware - Motherboard Overview

- Xilinx Zynq Z-7010[5] System on Chip with Dual Core ARM Cortex-A9 (PS) at 866MHz and Artix-7 FPGA section (PL)
  - 1Gbps Ethernet RJ-45
  - USB to Serial Converter
  - JTAG Connector
  - Uses Avent microZed[4] and FCI Connectors to reduce layout time.
    - Some power regulation done on board (1.8V and 3.3V)
    - Connectors part numbers Mouser #649-61082-101400LF, and #649-61083-101400LF.
- ADS62P49[3] 250MSPS Dual Channel 14-bit ADC
- 50Ω Single Ended Inputs and Outputs (SMA Connectors)
- Connector to 6 Channel DDS Board
- DC Barrel Jack to 12V 10A Laptop supply (Alternatives being inspected)
- Indicator LEDs (Error states, and some configuration info).
- GPIO and SPI breakout connector, for future add-ons.

Hardware - Motherboard ADC Selection

- Sampling Frequency: 250MSPS
- Resolution: 14 bits
- ENOB: 11.3 bits
- SNR: 73 dB
- INL_{max}: ±5 LSB
- DNL_{max}: ±1.3 LSB
- Input Range: $2V_{p-p}$
- Utilizing an undersampling method[6] to sample across a bandwidth from 50MHz to 500MHz.
- Phase and Amplitude error is minimal if frequencies in sweep are selected carefully.

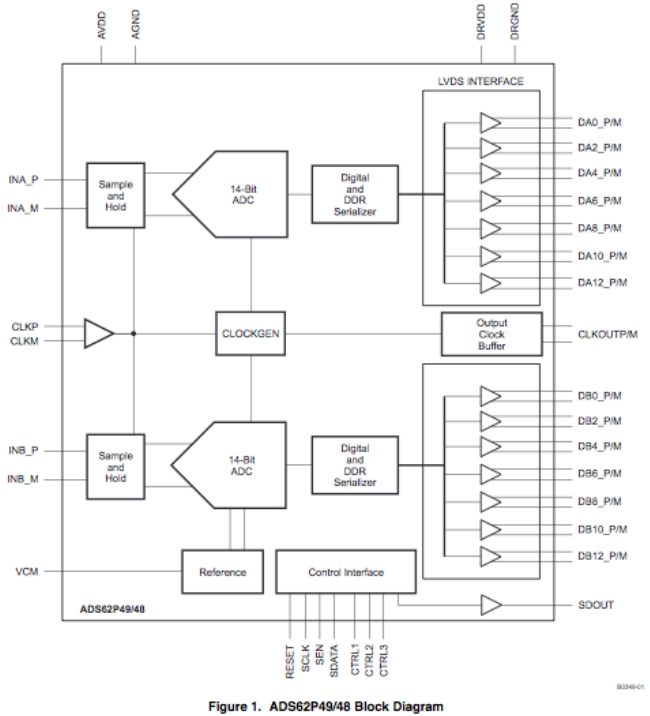


Figure 1. ADS62P49/48 Block Diagram

ADS62P49 Block Diagram[3]

Hardware - Motherboard ADC Selection Continued

- 7 1.8V CMOS Control Lines
- 1 LVDS pairs for each clock
  - Input and Output
- 14 LVDS Pairs for Data
  - 7 pairs per channel
  - posedge(clk) = EVEN
  - negedge(clk) = ODD

- Requires 1.8 V and 3.3 V
- Power at 250 MSPS 1.25 W

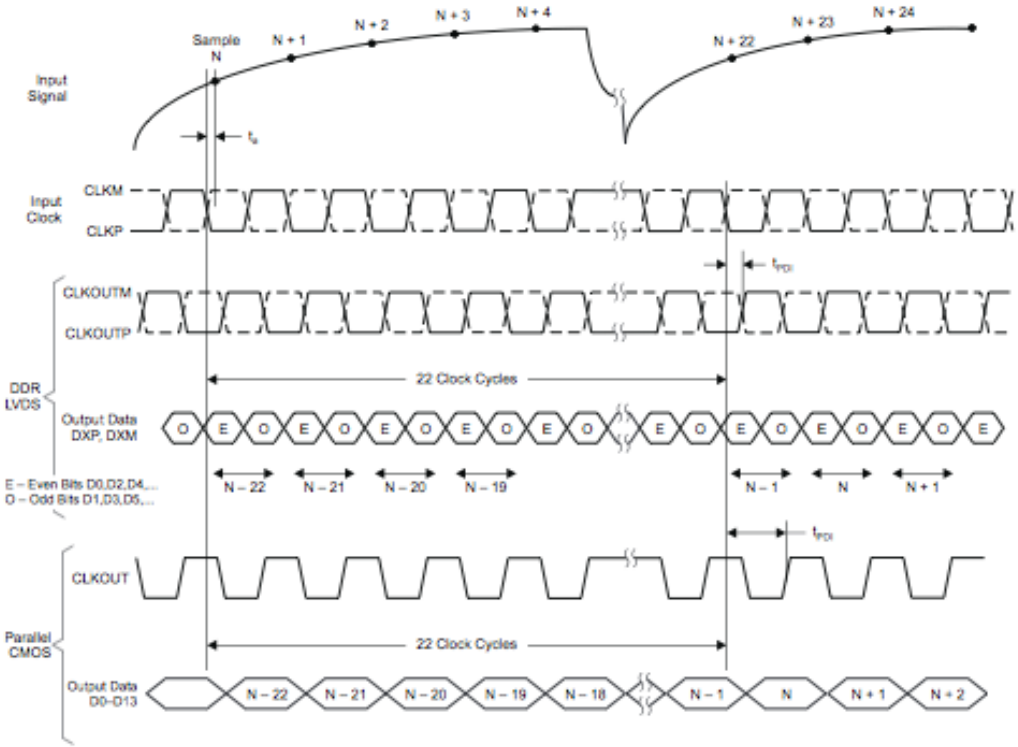
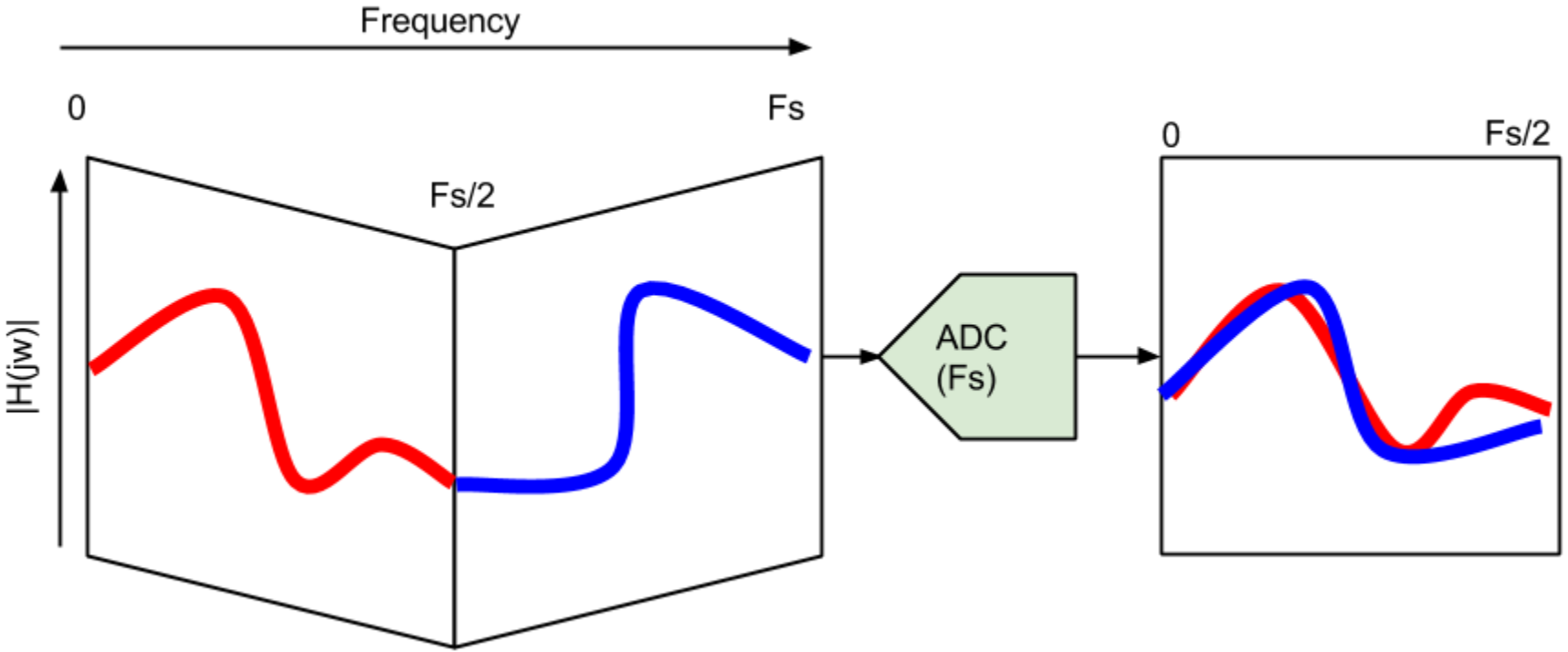


Figure 5. Latency Diagram

Hardware - A Defense of Undersampling

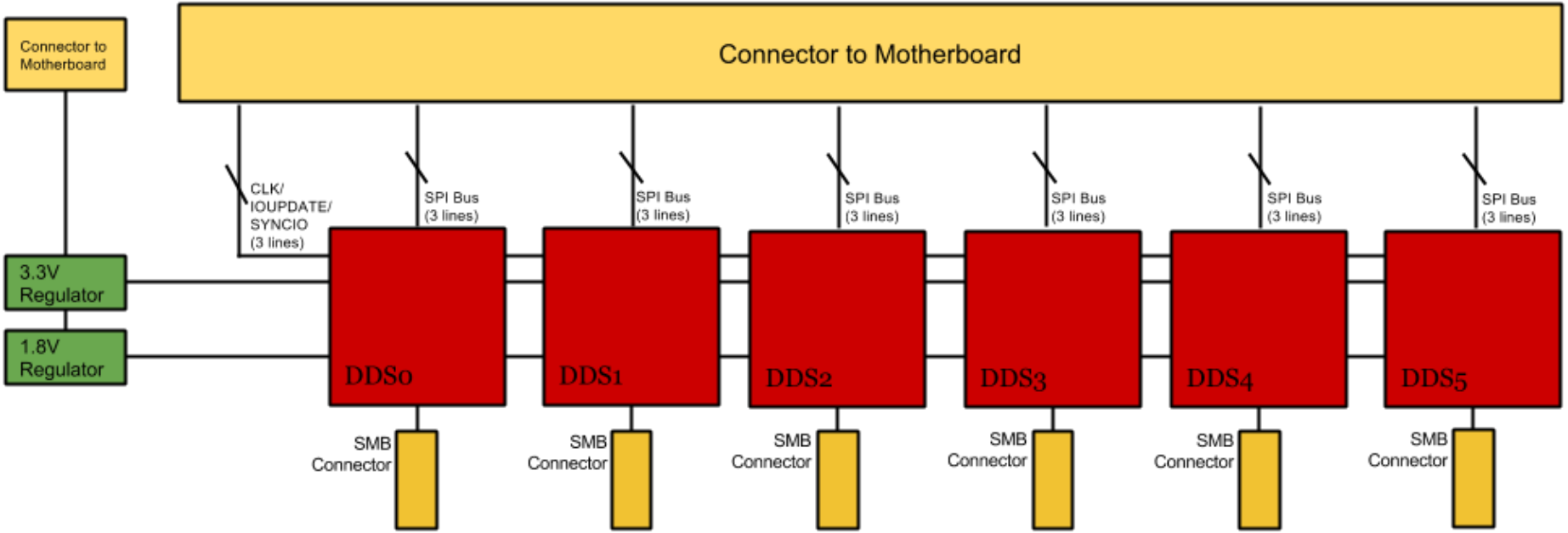


Hardware - A Defense of Undersampling

- Undersampling has been inspected by Justin Jung[6] (Researcher in Darren's Lab) and shown to have minimum phase and amplitude error.
  - Furthering the state of the art in terms of cost.

	250MHz	100MHz	50MHz
690nm	8.15% 0.076 °	9.04% 0.085 °	42.84% 0.241 °
785nm	2.15% 0.011 °	1.63% 0.015 °	30.24% 0.133 °
830nm	4.79% 0.026 °	3.99% 0.027 °	24.35% 0.202 °
850nm	3.53% 0.023 °	3.01% 0.028 °	32.09% 0.180 °

Hardware - Frequency Synthesis Board Block Diagram



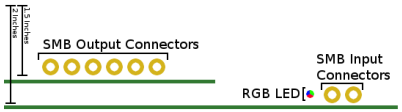
Hardware - Frequency Synthesis Board

- 6 DDS chips (AD9914[3])
- 3.5 GSPS 12 bit DAC
- Outputs to SMB connectors passed out of the enclosure
- Each chip is on an independant SPI bus
- Each chip needs #CS, SDIO, and SDO pins
- All chips serialy connected to SCLK, IOUPDATE, and SYNCIO
- All chips need 1.8V and 3.3V for both AVDD and DVDD
- Power will be regulated from 12V on board
- 5V and SPI communication brought from motherboard over ribbon cable connector

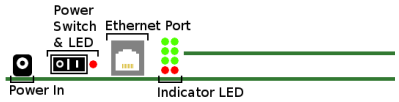
Hardware - Enclosure

Size:

9.32” x 6.18” x 3.15”

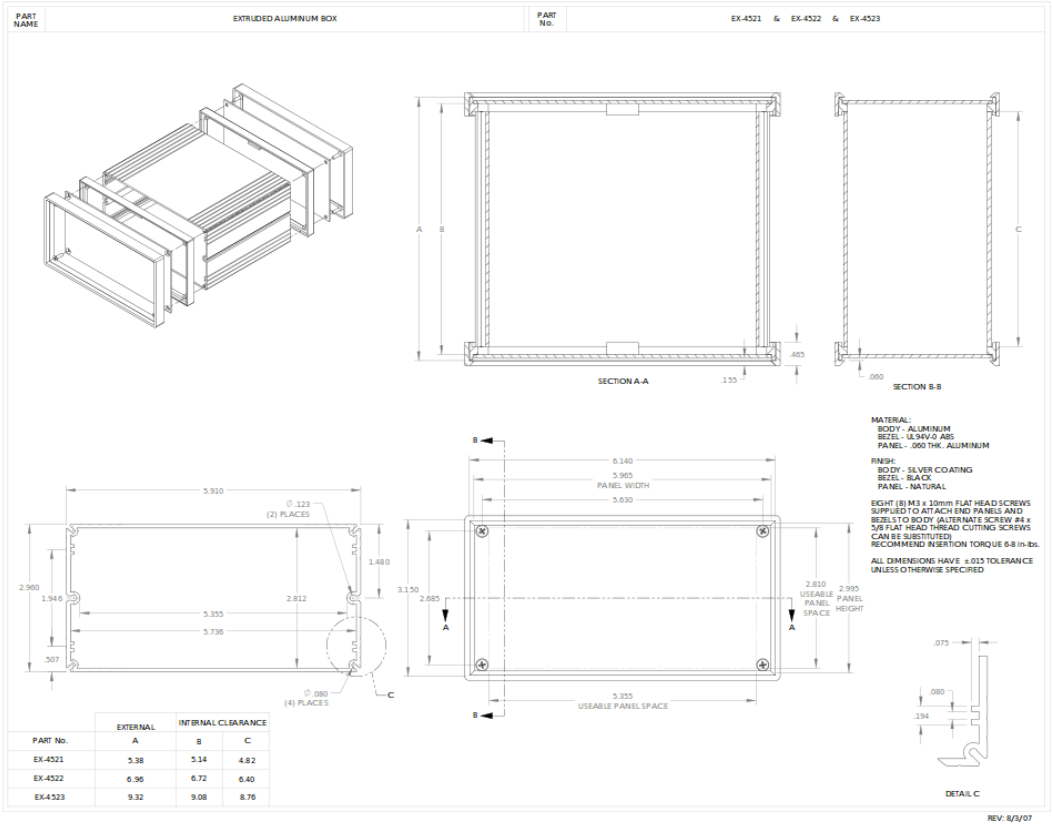


Custom Face Plate



Professional Look

prospective enclosure mechanical specs [7]



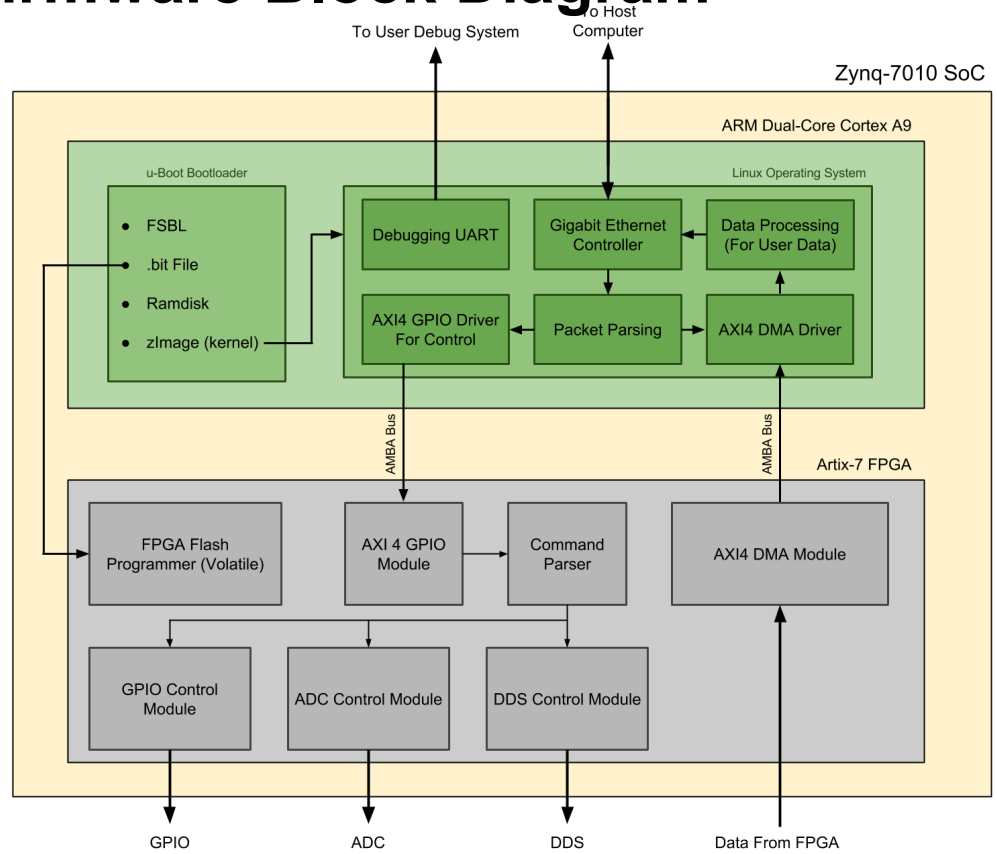
PART No.	EXTERNAL A	INTERNAL CLEARANCE B	C
EX-4521	5.38	5.14	4.82
EX-4522	6.96	6.72	6.40
EX-4523	9.32	9.08	8.76

Power Regulation

- Off the shelf 12V 10A wall power regulator
- Motherboard switching regulator to 5V
- Motherboard voltage requirements: 1.8V - 3.3V - 5V
  - microZed - 5V
  - DDS connector - 5V
  - ADC Chip - 1.8V - 3.3V
- DDS board voltage requirements: 1.8V - 3.3V
  - DDS Chip - 1.8V - 3.3V

Firmware Design

Firmware Block Diagram



Firmware Specification Overview

Hardware: The firmware will run on a microZed (µZed). The µZed has the following features:

- Zynq Z010 SoC Development Board
  - ARM Dual Core A9 Processor
  - Artix 7 FPGA
- 1GB DDR3 SDRAM
- 128Mb QSPI Flash Memory
- 10/100/1000 Ethernet
- USB 2.0, USB-UART
- 100 User IO w. 48 LVDS capable pairs (using an outside header breakout)

Firmware Processing System Modules

uBoot Bootloader: Initializes devices on the Zynq as well as peripherals, programs PL in FPGA, (must be done every power cycle due to volatile nature of FPGA) and loads the Linux kernel

UART Debugger: Minimalistic testing module to allow for customer debugging via the UART cable.

Gigabit Ethernet Controller: Receives command data from host computer and transmits collected ADC data.

Packet Parsing: Parses received data from the host and sends control signals to the AXI modules (the “boss” of the firmware)

Firmware Processing System Modules

AXI4 GPIO Driver: Sends user commands to the DDS, ADC, and GPIO PL Modules. Includes a general interface to the GPIO for future system modifications implemented by the user

AXI4 DMA Driver: Gets the ADC data from SDRAM. There will be a protected portion of memory that this data will be wrote to

Data Processing Module: Takes the data from the DMA driver to process and place into a packet for transmission to the host computer via gigabit ethernet

Firmware Programmable Logic Modules

FPGA Flash Programmer: Module is integrated into bootloader design workflow, programs the PL at power on.

AXI4 GPIO Module: Receives commands from the PS

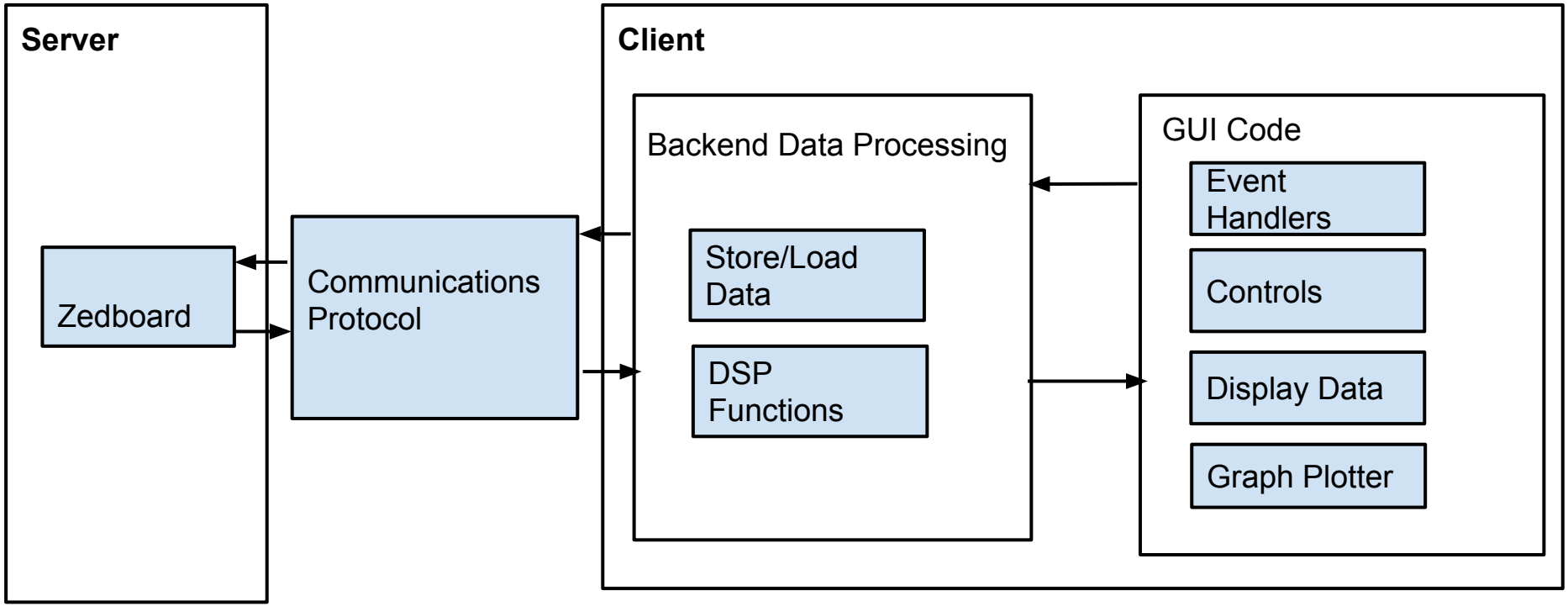
Command Parser: Parses the GPIO commands

DDS/ADC/GPIO Control: Sends control signals to the DDS, ADC, and GPIO, respectively.

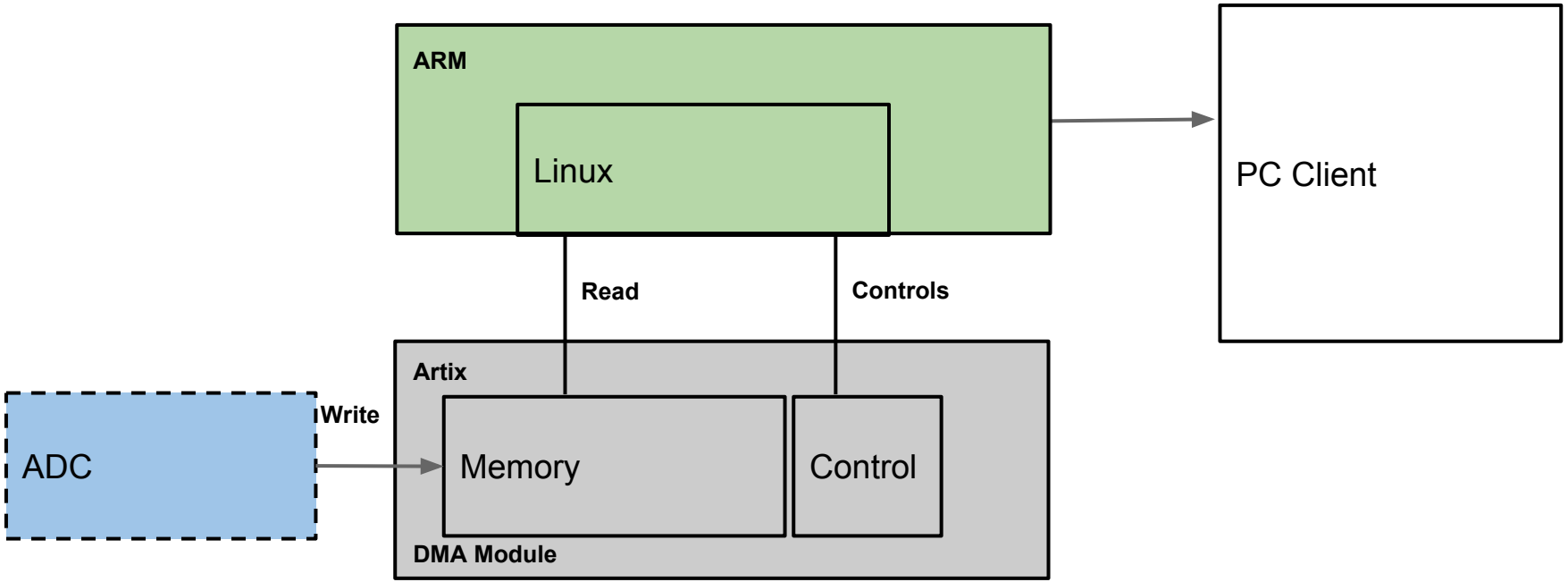
AXI4 DMA Module: Receives data from the ADC and then sends it directly (without processing) to the peripheral SDRAM. It does this over the AMBA bus at up to 950 Mbps

Software Design

Software - Block Diagram



Software - Data Flow



Software - Networking

$$Sweep\ Size = (\#\ of\ Steps) * (\#\ of\ Samples) * (\#\ of\ Channels) * (Sample\ Size)$$

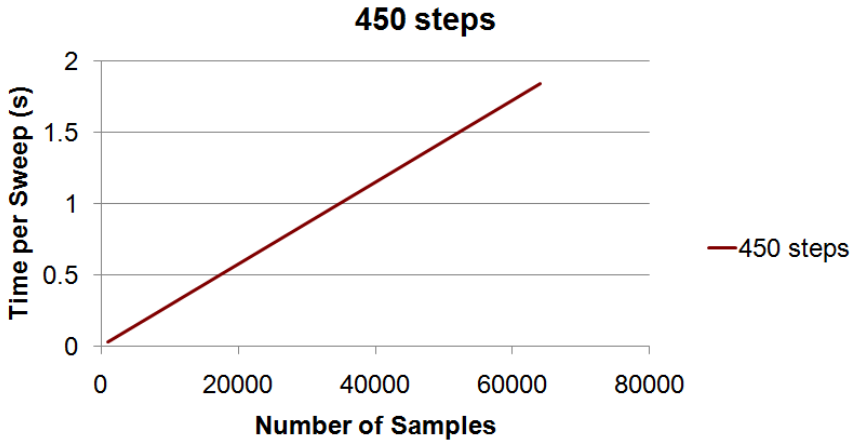
$$Rate\ of\ data\ transfer = efficiency * transfer\ rate\ of communication\ link$$

$$Time\ per\ sweep = \frac{Sweep\ Size}{Rate\ of\ data\ transfer}$$

$$Data\ Aquisition\ Rate = \ Size\ per\ Sweep * Sweeps\ per\ second$$

Software - Networking

Assuming 50% gigabit link transmission efficiency



Nominal Case
Samples per step: 4000
Number of steps: 450

Size = 6.86 MB/Sweep

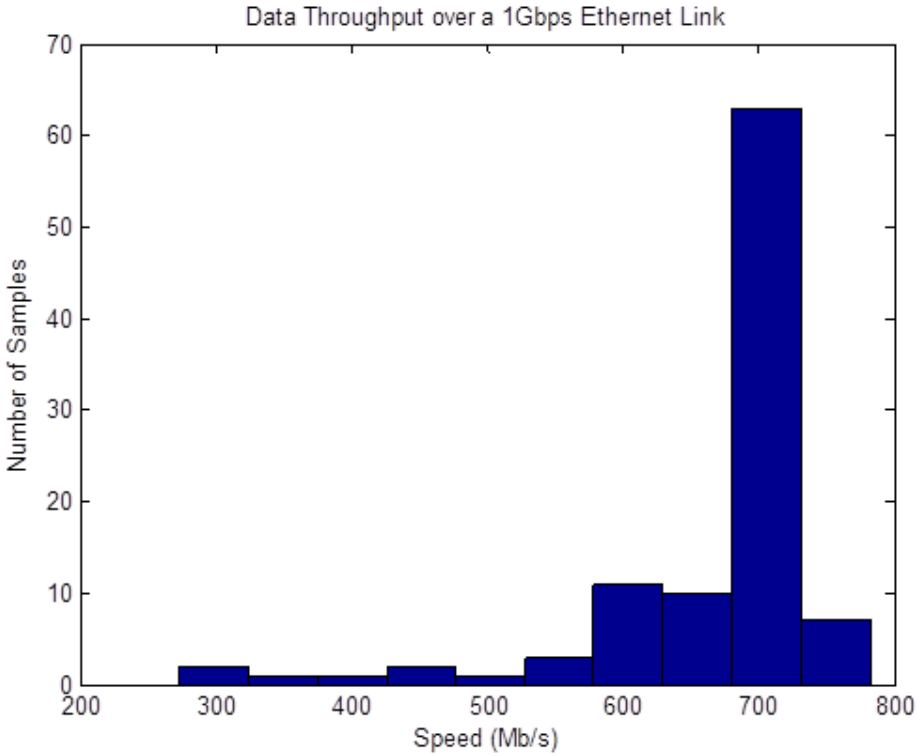
Worst Case
Samples per step: 64000
Number of steps: 450

Size = 118 MB/Sweep

$$Sweep\ Size = (\#\ of\ Steps) * (\#\ of\ Samples) * (\#\ of\ Channels) * (Sample\ Size)$$

$$Time\ per\ sweep = \frac{Sweep\ Size}{Rate\ of\ data\ transfer}$$

Performance Testing TCP



Sending a file over 1Gbps link.

No transmission errors encountered.

Histogram of Test. Results show an average efficiency of 70%.

Software - Protocols

TCP + Roll Your Own Data Protocol

- Can be extremely efficient with minimal overhead
- Build on top of TCP.
- Need to implement socket and protocol layers ourselves.

ZeroMQ

- Build on top of TCP and wraps sockets.
- A lot of development time already put into efficiency.
- Technically has more overhead than a hand crafted protocol.

Software - Protocols

TCP

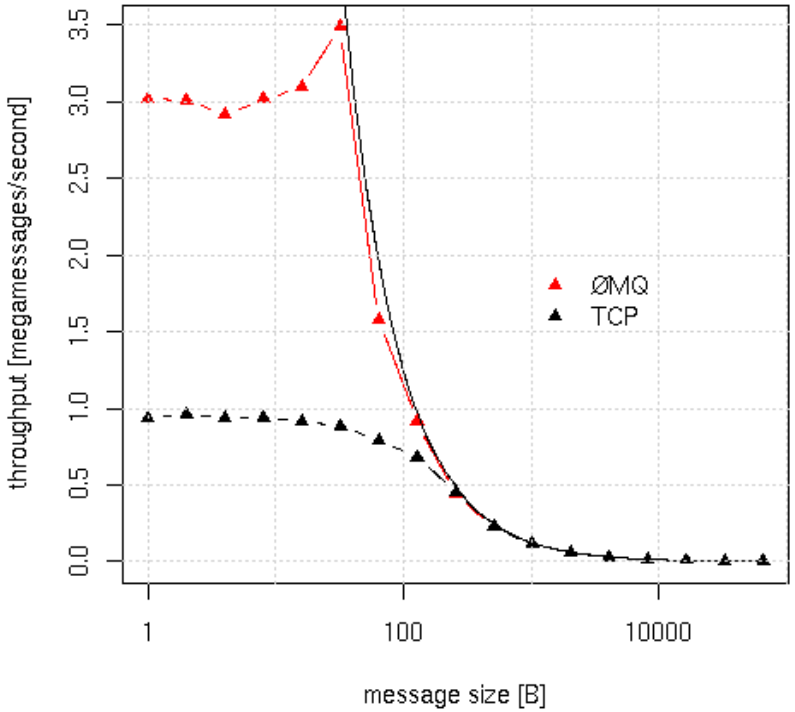
- Very efficient on low latency high bandwidth networks
- Need to implement socket and protocol layers ourselves.

ZeroMQ

- Build on top of TCP and wraps sockets.
- A lot of development time already put into efficiency.
- Technically has more overhead than a hand crafted protocol.

Software - Protocols

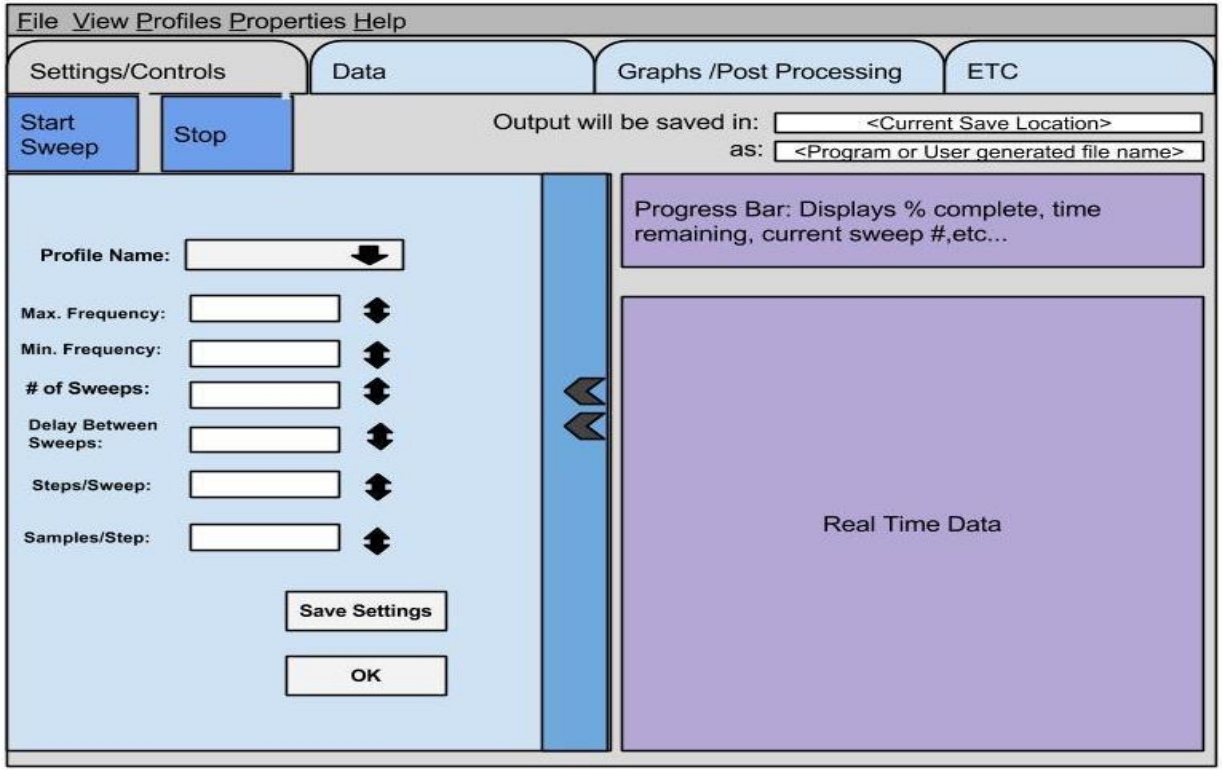
Software - Protocols (ZMQ vs TCP)



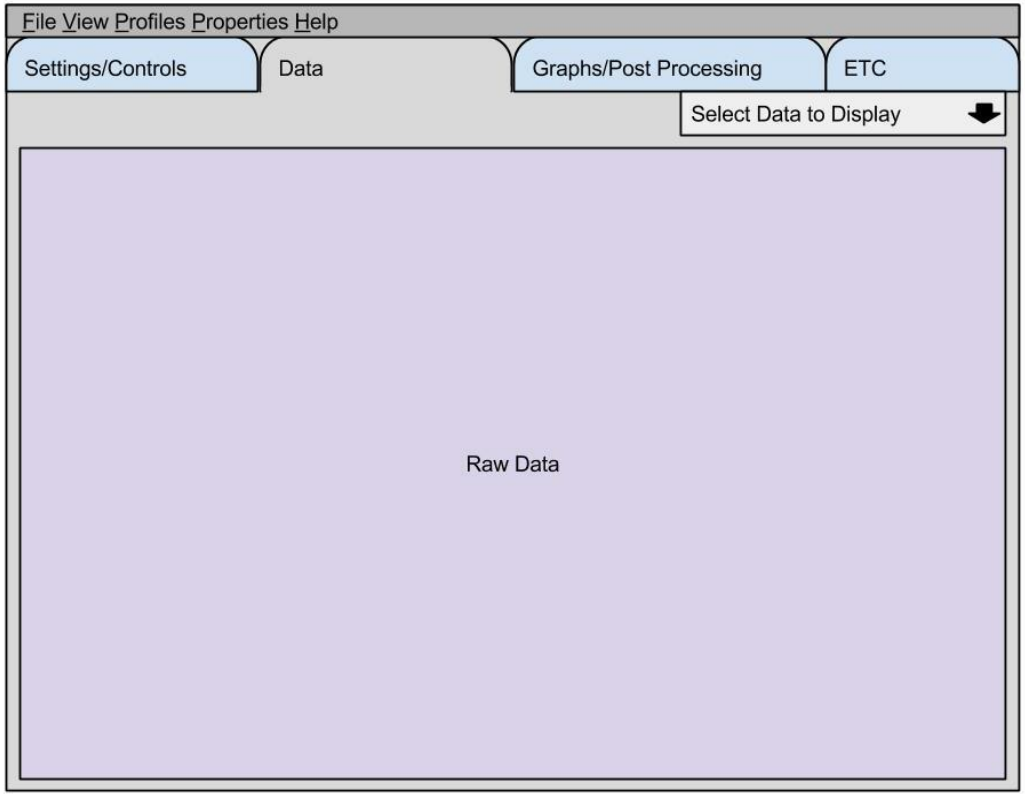
Software - GUI Back End

- Must be able to:
    - Allow user to set controls
    - Load/Store Data
    - Extract phase and amplitude information
    - Graph relevant information
    - DSP functions (possibly)
- Create DLL to carry out functionality

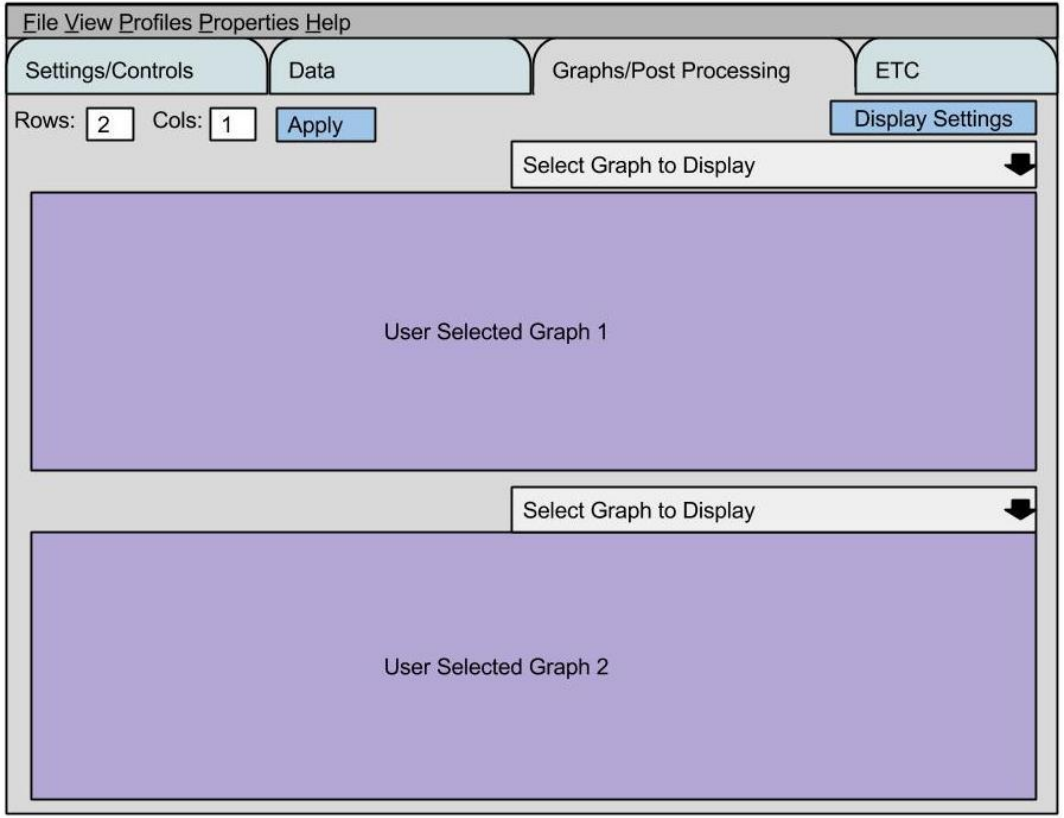
Software - GUI



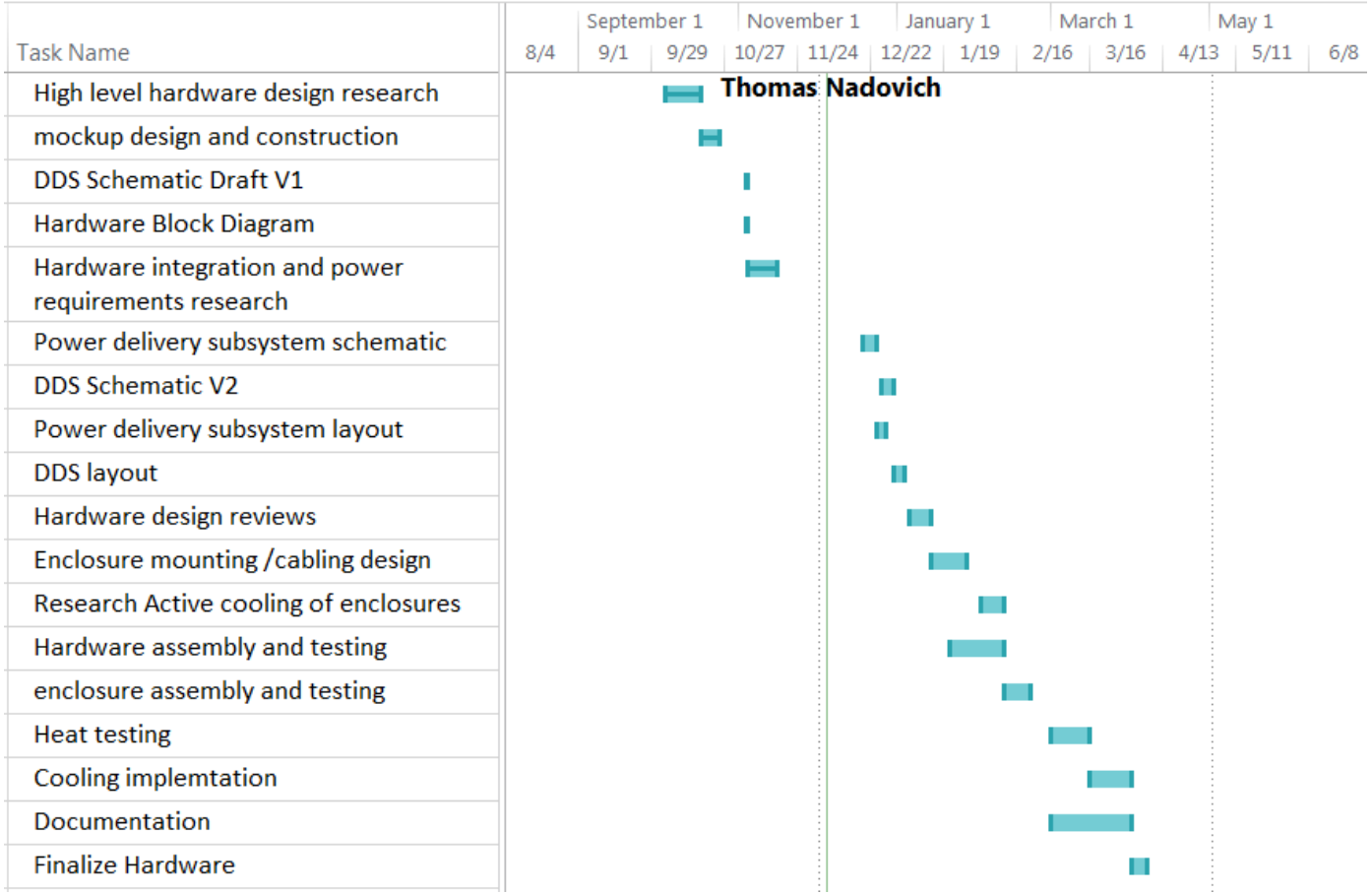
Software - GUI

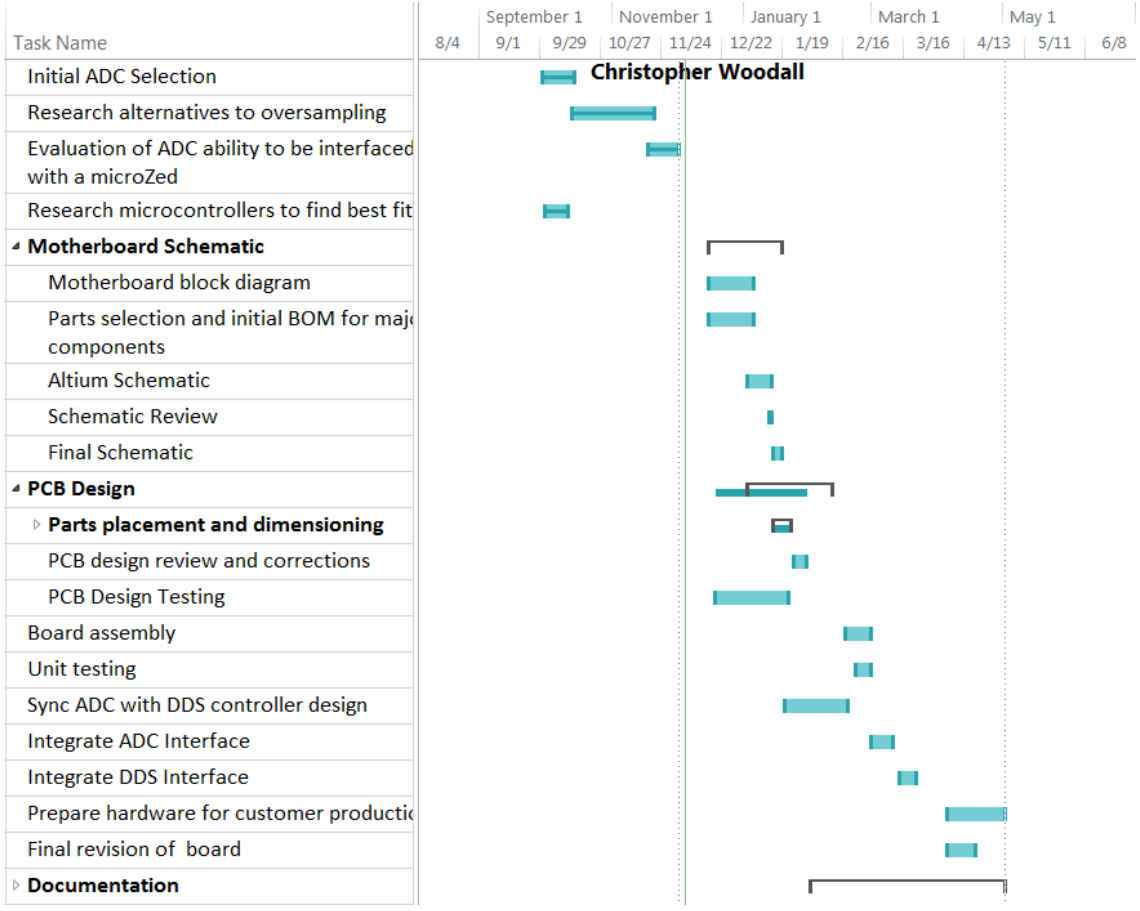


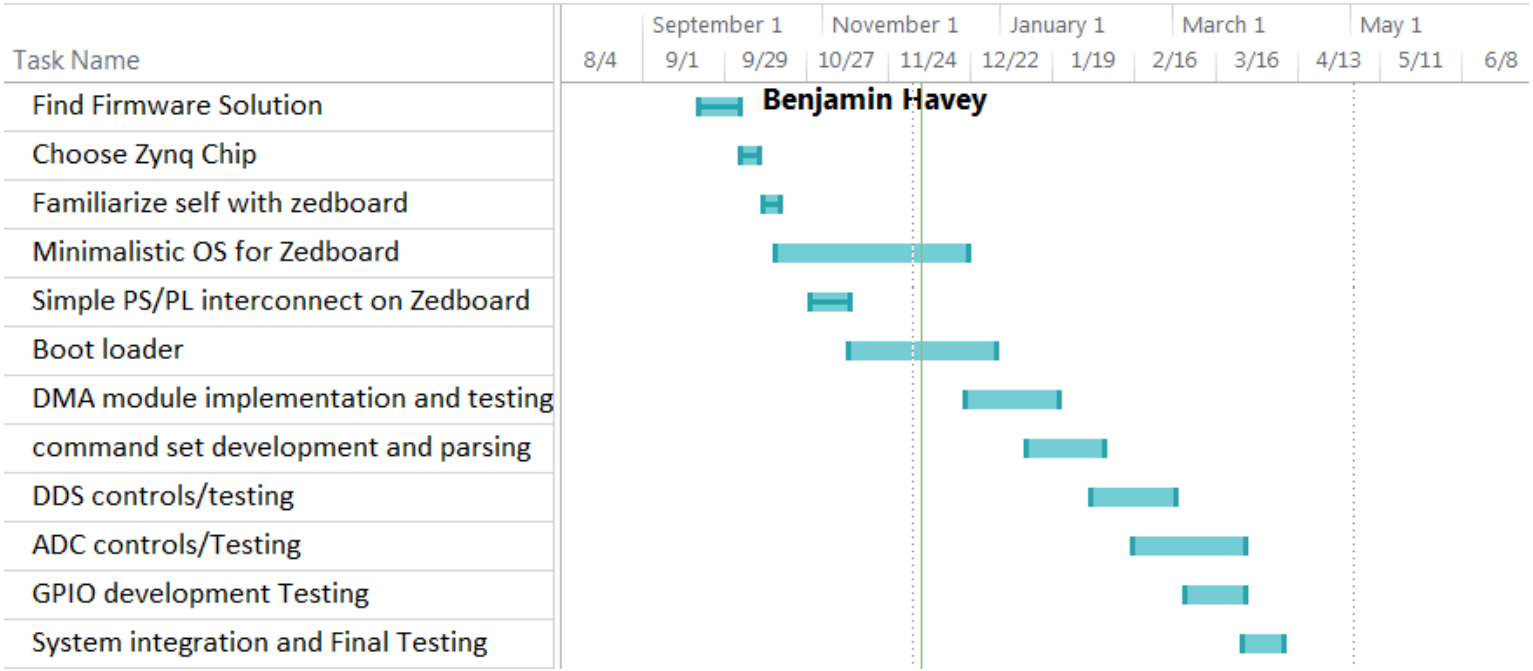
Software - GUI

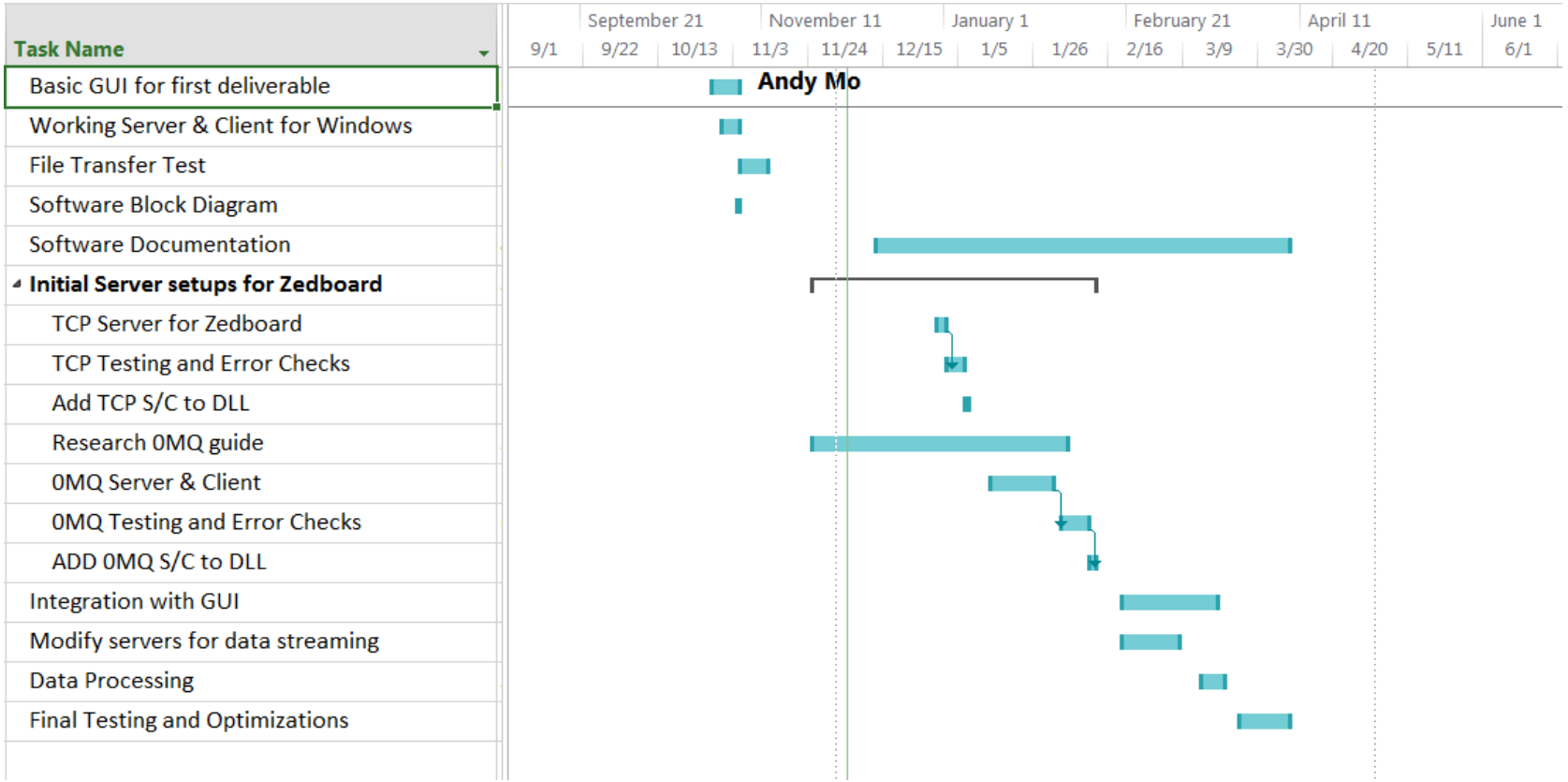


Schedules

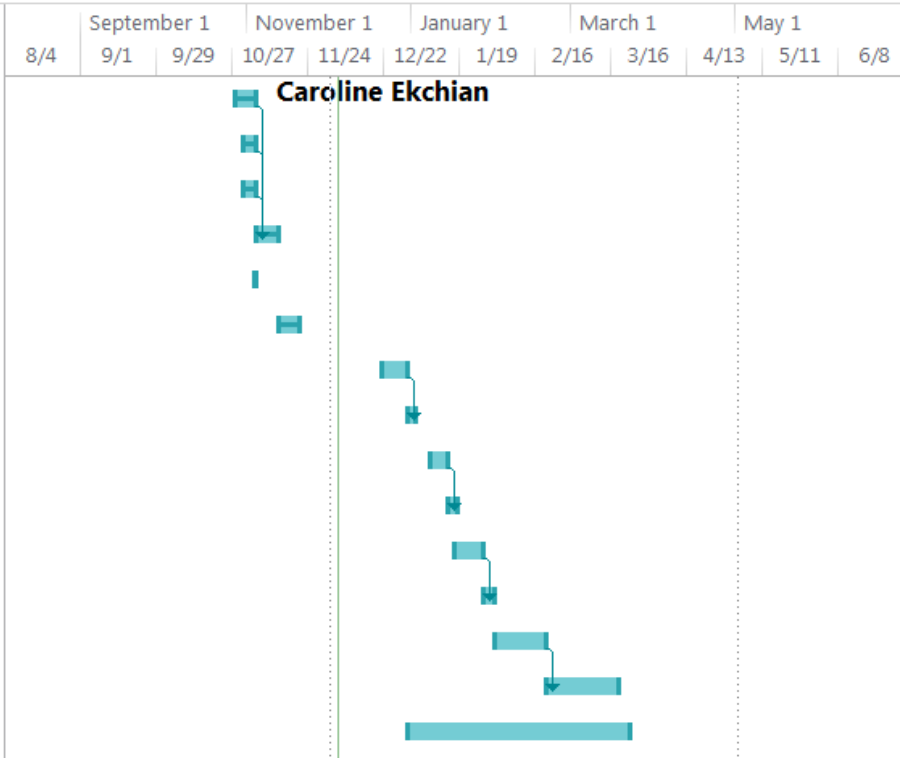

Task Name
September 1
November 1
January 1
March 1
May 1
8/4
9/1
9/29
10/27
11/24
12/22
1/19
2/16
3/16
4/13
5/11
6/8
High level hardware design research
mockup design and construction
DDS Schematic Draft V1
Hardware Block Diagram
Hardware integration and power requirements research
Power delivery subsystem schematic
DDS Schematic V2
Power delivery subsystem layout
DDS layout
Hardware design reviews
Enclosure mounting /cabling design
Research Active cooling of enclosures
Hardware assembly and testing
enclosure assembly and testing
Heat testing
Cooling implemtation
Documentation
Finalize Hardware
Thomas Nadovich


Task Name
September 1
November 1
January 1
March 1
May 1
8/4
9/1
9/29
10/27
11/24
12/22
1/19
2/16
3/16
4/13
5/11
6/8
Christopher Woodall
Initial ADC Selection
Research alternatives to oversampling
Evaluation of ADC ability to be interfaced with a microZed
Research microcontrollers to find best fit
Motherboard Schematic
Motherboard block diagram
Parts selection and initial BOM for majo components
Altium Schematic
Schematic Review
Final Schematic
PCB Design
Parts placement and dimensioning
PCB design review and corrections
PCB Design Testing
Board assembly
Unit testing
Sync ADC with DDS controller design
Integrate ADC Interface
Integrate DDS Interface
Prepare hardware for customer productio
Final revision of board
Documentation


Task Name
September 1
November 1
January 1
March 1
May 1
8/4
9/1
9/29
10/27
11/24
12/22
1/19
2/16
3/16
4/13
5/11
6/8
Find Firmware Solution
Benjamin Havey
Choose Zynq Chip
Familiarize self with zedboard
Minimalistic OS for Zedboard
Simple PS/PL interconnect on Zedboard
Boot loader
DMA module implementation and testing
command set development and parsing
DDS controls/testing
ADC controls/Testing
GPIO development Testing
System integration and Final Testing


Task Name
Basic GUI for first deliverable
Working Server & Client for Windows
File Transfer Test
Software Block Diagram
Software Documentation
Initial Server setups for Zedboard
TCP Server for Zedboard
TCP Testing and Error Checks
Add TCP S/C to DLL
Research 0MQ guide
0MQ Server & Client
0MQ Testing and Error Checks
ADD 0MQ S/C to DLL
Integration with GUI
Modify servers for data streaming
Data Processing
Final Testing and Optimizations
September 21
November 11
January 1
February 21
April 11
June 1
9/1
9/22
10/13
11/3
11/24
12/15
1/5
1/26
2/16
3/9
3/30
4/20
5/11
6/1
Andy Mo

Task Name	Caroline Ekchian
Basic GUI for first deliverable	
Working Client in C++	
Working Server in C++	
File Transfer Test	
Software Block Diagram	
First Deliverable Test	
Settings/Controls Tab	
Add setting / Control Functionality to DLL	
Graphing	
Add graphing functionality to DLL	
Post Processing	
Add Post Processing to DLL	
Testing GUI with another PC as a server	
Integration/Testing/Debuging with zedbo	
Software Documention	



References

[1]: Uedoa, Shigeto, Darren Roblyer, et al. “Baseline Tumor Oxygen Saturation Correlates with a Pathologic Complete Response in Breast Cancer Patients Undergoing Neoadjuvant Chemotherapy”. Cancer Research. July 8, 2012.

[2]: Texas Instruments. “ADS62P49 Datasheet”. January 2011.
http://www.ti.com/lit/ds/symlink/ads62p49.pdf

[3]: Analog Devices. “AD9910 Datasheet”. May 2012.
http://www.analog.com/static/imported-files/data_sheets/AD9914.pdf

[4]: Avnet. “Zedboard Hardware User Guide”. November 2013.
http://www.zedboard.org/sites/default/files/documentations/MicroZed_HW_UG_v1_2.pdf

[5]: Xilinx. “XC7Z010 Datasheet”.
http://www.xilinx.com/support/documentation/data_sheets/ds187-XC7Z010-XC7Z020-Data-Sheet.pdf

[6]: Justin Jung, and Darren Roblyer. “Feasibility of Undersampling”. Email. November 2013.

[7]: Bud Industries. “Prospective enclosure Datasheet”. August 2007.
http://www.budind.com/pdf/hb4521.pdf

[8]: Roblyer, Darren, et al. “Feasibility of direct digital sampling for diffuse optical frequency domain spectroscopy in tissue”. Meas. Sci. Technol. 24. 2013.